

EXHIBIT 19



FPGA Architecture

Introduction

Altera continues to lead the FPGA industry in architectural innovation. The logic fabric and routing architecture in Altera® FPGAs are unmatched, providing customers with a number of advantages. Altera was the first to introduce the 8-input fracturable look-up table (LUT) with the Stratix® II family in 2004. At its core is the adaptive logic module (ALM) with 8 inputs, which can implement a full 6-input LUT (6-LUT) or select 7-input functions. The ALM can also be efficiently partitioned into independent smaller LUTs, providing the performance advantage of larger LUTs and the area efficiency of smaller LUTs. The Stratix series of FPGAs also excels in routing through the MultiTrack™ interconnect, which provides the industry's best connectivity. As a result, Altera FPGA architecture is at least one generation ahead of the competition, and routing architecture is two generations ahead.

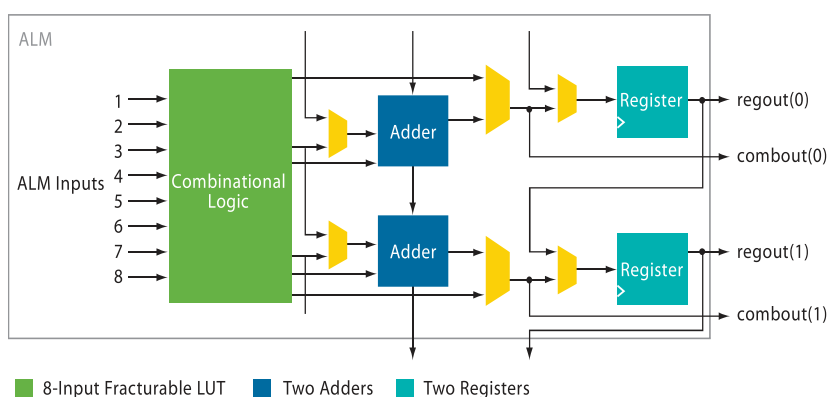
This paper describes the leading-edge architectural innovations in Altera FPGAs and their advantages:

- The ALM's 1.8X density advantage over the competition
- Optimal register-to-logic ratio (2:1) to ensure that the devices are not register-limited
- The most routing connectivity, with up to five times the logic in a single hop compared to the competition

Logic Fabric

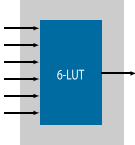
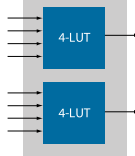
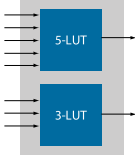
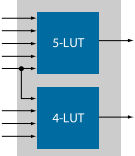
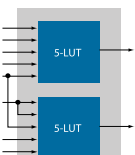
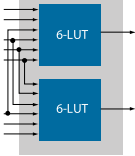
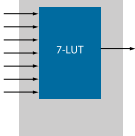
The key to the high-performance, area-efficient architecture is the ALM. It consists of combinational logic, two registers, and two adders as shown in Figure 1. The combinational portion has eight inputs and includes a LUT that can be divided between two adaptive LUTs (ALUTs) using Altera's patented LUT technology. An entire ALM is needed to implement an arbitrary six-input function, but because it has eight inputs to the combinational logic block, one ALM can implement various combinations of two functions.

Figure 1. Adaptive Logic Module (ALM) Block Diagram



In addition to implementing a full 6-input LUT, the ALM can, for example, implement 2 independent 4-input functions or a 5-input and a 3-input function with independent inputs. Table 1 shows a summary of combinational logic configurations supported in an ALM. For a more detailed architectural description, refer to the *Stratix II Device Handbook*. Because 2 registers and 2 adders are available, the ALM has the flexibility to implement 2.5 logic elements (LEs) of a classic 4-input LUT (4-LUT) architecture, consisting of a 4-LUT, carry logic, and a register.

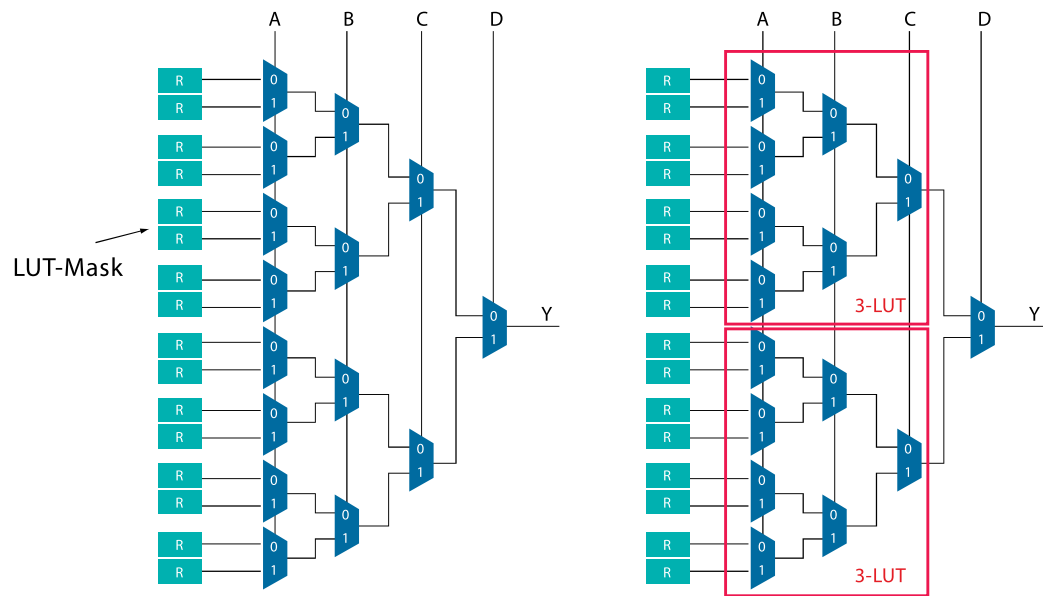
Table 1. ALM Flexibility

Configuration	Description
	One Stratix II ALM can input any 6-input function.
	One Stratix II ALM can be configured to implement 2 independent 4-input or smaller LUTs. This configuration can be viewed as the “backward-compatibility” mode. Designs that are optimized for the traditional 4-LUT FPGAs can easily be migrated to the Stratix II family.
	One Stratix II ALM can be configured to implement a 5-LUT and 3-LUT. The inputs to the two LUTs are independent of each other. The 3-LUT can be used to implement any logic function that has 3 or fewer inputs. Therefore, a 5-LUT/2-LUT combination is also available.
	One Stratix II ALM can be configured to implement a 5-LUT and a 4-LUT. One of the inputs is shared between the 2 LUTs. The 5-LUT has up to 4 independent inputs. The 4-LUT has up to 3 independent inputs. The sharing of inputs between LUTs is very common in FPGA designs, and the Quartus® II software automatically seeks logic functions that are structured in this manner.
	One Stratix II ALM can be configured to implement two 5-LUTs. Two of the inputs between the LUTs are common, and up to 3 independent inputs are allowed for each 5-LUT.
	<p>If two 6-input functions have the same logic operation and 4 shared inputs, the two 6-input functions can be implemented in one Stratix II ALM.</p> <p>For example, a 4x2 crossbar switch with 4 data input lines and 2 sets of unique select signals requires four LEs in the Stratix family. In the Stratix II family, this function only requires one ALM. Another example is a 6-input AND gate. An ALM can implement two 6-input AND gates that have 4 common inputs. The same function would require 3 LEs if implemented in a Stratix device.</p>
	One Stratix II ALM in the extended mode can implement a subset of a 7-variable function. The Quartus II software automatically recognizes the applicable 7-input function and fits it into an ALM. Refer to the <i>Stratix II Device Handbook</i> for detailed information about the types of 7-input functions that can be implemented in an ALM.

Building Look-up Tables (LUTs)

An overview of how LUTs are built helps describe the key innovations in the ALM. A LUT is typically built out of SRAM bits to hold the configuration memory (CRAM) LUT-mask and a set of multiplexers to select the bit of CRAM that is to drive the output. To implement a k-input LUT (k-LUT)—a LUT that can implement any function of k inputs— 2^k SRAM bits and a $2^k:1$ multiplexer are needed. **Figure 2** shows a 4-LUT, which consists of 16 bits of SRAM and a 16:1 multiplexer implemented as a tree of 2:1 multiplexers. The 4-LUT can implement any function of 4 inputs (A, B, C, D) by setting the appropriate value in the LUT-mask. To simplify the 4-LUT in **Figure 2**, it can also be built from two 3-LUTs connected by a 2:1 multiplexer.

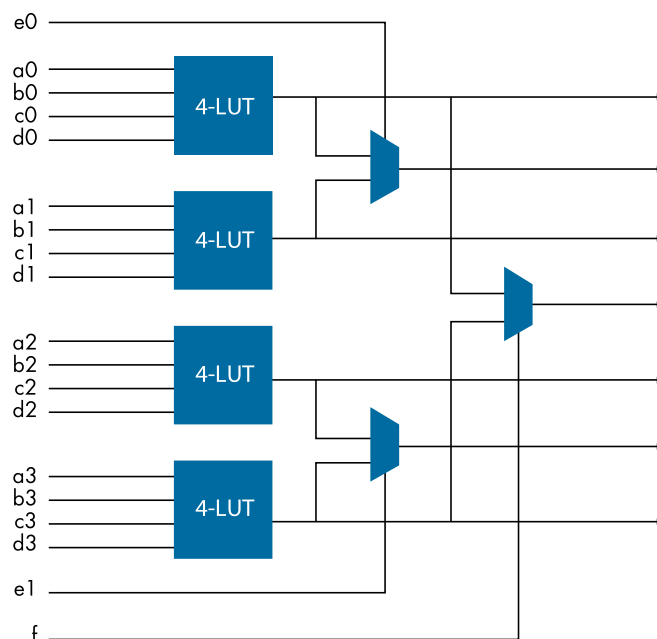
Figure 2. Building a LUT



$$a'b'c'd' + abcd + abc'd' = 1000\ 0000\ 0000\ 1001 = 0x8009$$

Similarly, larger LUTs can be built out of smaller ones, as shown in **Figure 3**. For example, a 5-LUT can be built with two 4-LUTs and a multiplexer, while a 6-LUT can be built with two 5-LUTs and a multiplexer. Technically, what matters is the total number of CRAM bits in the LUT and that they are used to implement an arbitrary function of six inputs.

Figure 3. Composing Larger LUTs from Smaller LUTs

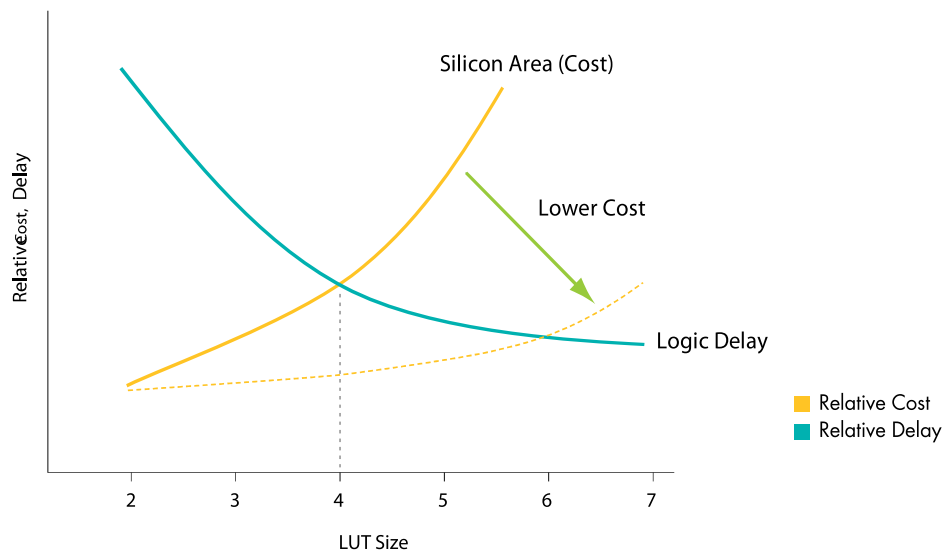


Even though larger LUTs can be built from smaller LUTs, it is important to differentiate between FPGA architectures designed for 4-LUTs and for 6-LUTs. With a different size LUT as the base logic block, the number of LUTs clustered together in each architecture, the number of inputs available to the LUTs, and delay optimization through the LUTs will vary. While 6-LUTs can be built on architectures that support 4-LUTs, this structure is inefficient. For example, four 4-LUTs together with either a 4:1 multiplexer or 2 more 4-LUTs can be used to build a 6-LUT as shown in Figure 3, but the implementation uses only 6 of the 16 available inputs and creates extra delays between the various LUTs. Clearly, having the ability to build 6-input LUTs is not enough; the entire architecture needs to be optimized specifically for 6-LUTs as the base logic block. Altera was the first to offer an architecture optimized for 6-LUT performance with the Stratix II FPGA family.

Designing the ALM

The ALM is radically different from any other FPGA logic block, offering a number of major innovations. Getting from a classic 4-LUT with a single register block (with associated carry logic) to the ALM required a detailed understanding of customer requirements and a large investment in researching the tradeoffs of various architectures. Our pursuit for a larger LUT was inspired by research results indicating that a basic 6-LUT could yield a 14% performance improvement by reducing the number of levels of logic elements on the critical paths of circuits. Unfortunately, this performance increase also had a large area penalty, a 17% area increase resulting from a larger LUT-mask and more inputs for the LUT. Figure 4 illustrates the tradeoff between cost and delay for different sizes of LUTs. The basic approach in designing the ALM was to investigate building a larger LUT to reduce levels of logic and increase performance, but to also avoid the area increase by efficiently dividing the larger LUT into smaller LUTs when appropriate, as illustrated by the dashed line. The ability to divide a LUT is what makes it “adaptive.”

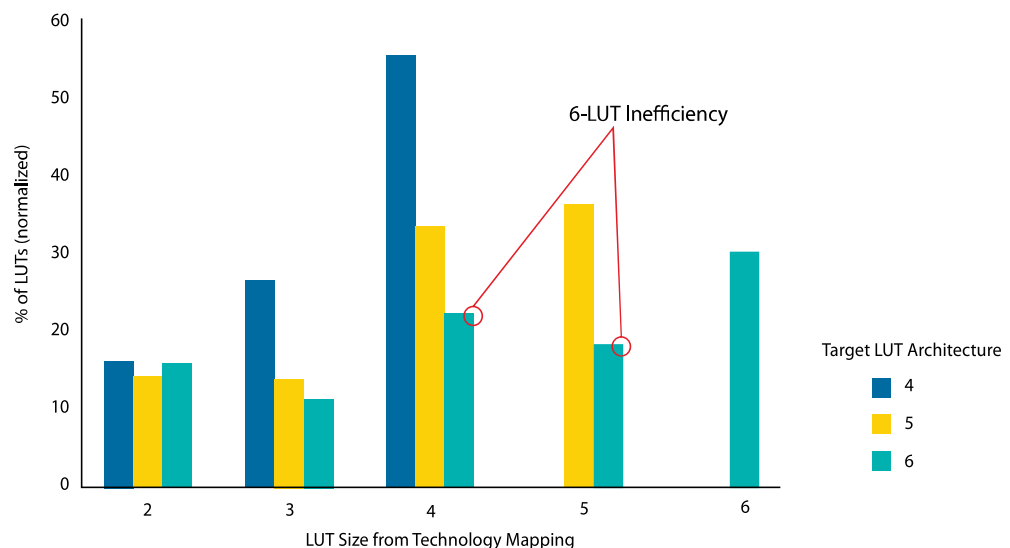
Figure 4. Delay-Cost Tradeoff with LUT Size



Investigations into alternative designs for the adaptive LUT began by using a proprietary tool that models the complex interaction of FPGA hardware and software, allowing Altera to explore different architectural implementations. The tool models the details of an FPGA design, various logic and memory resources, a hierarchical description of the architecture, and all of the necessary FPGA timing and physical details. When designing a new logic block architecture, rather than just incrementally improving an existing one, it is critical to be able to model the block itself in terms of LUT size and logic element features, and the software that targets it. In addition to the research place-and-route tools, Altera uses a research synthesis tool to synthesize and pack for the target logic block. The results of many designs are analyzed for speed, area, routability, and power.

To better understand the area penalty associated with a basic 6-LUT, Altera analyzed the number of inputs generated for each logic function. Figure 5 shows the distribution of LUT sizes produced during synthesis when targeting basic 4-, 5-, and 6-LUT architectures.

Figure 5. Distribution of Functions Generated by Synthesis



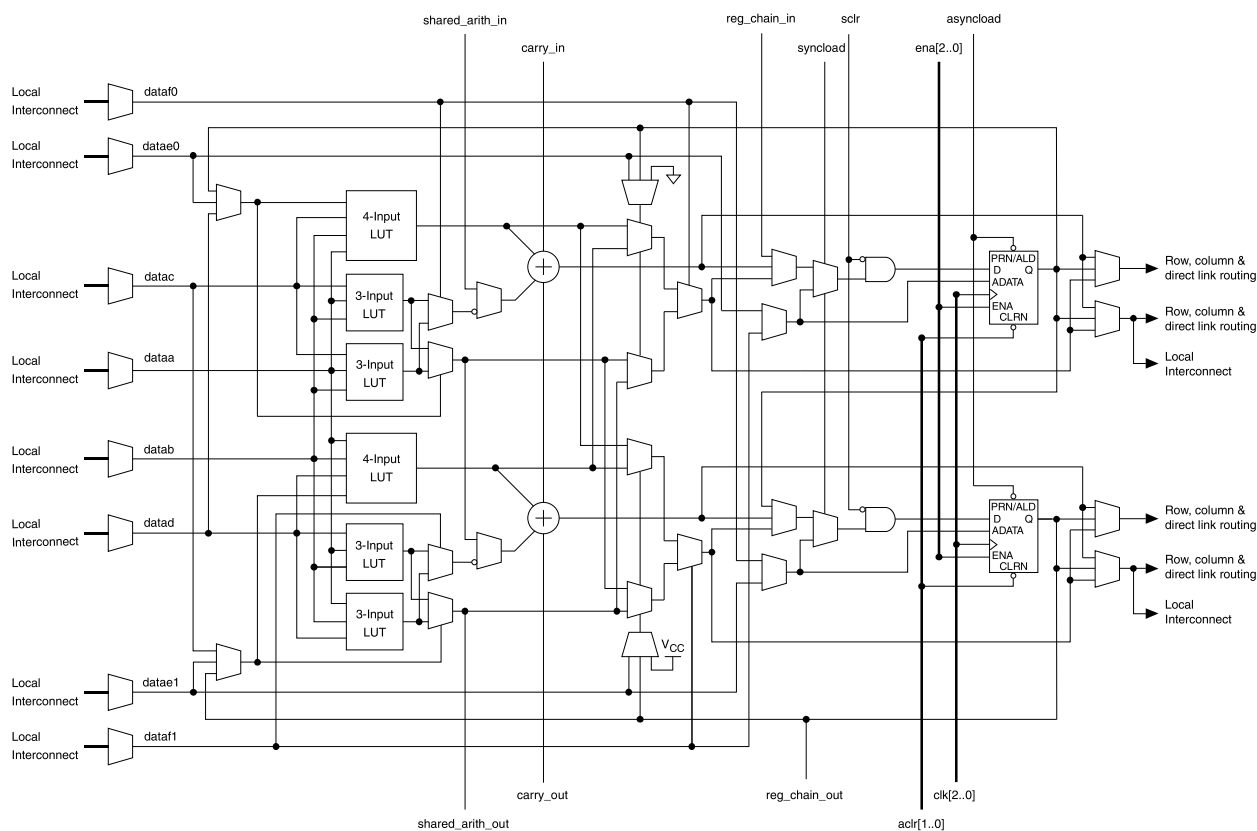
Focusing on the speed optimization results, the key findings include:

- A mix of LUT sizes are produced, and the majority of functions are not 6-LUTs. Even though 6-LUTs are being targeted, only about 30% can be extracted in this experiment because there are fewer large cones of logic in circuits to be absorbed into single functions.
- Smaller functions are not implemented efficiently in an architecture that only supports basic 6-LUTs because configuration bits are unused, wasting silicon area and increasing cost. This inefficiency can be avoided if the 6-LUT can be divided into smaller LUTs so that more than one smaller function can be implemented in the 6-LUT resource.

In the search for a larger LUT that offers performance benefits as well as the ability to efficiently implement smaller functions, many ideas were investigated. Composing a 6-LUT out of 4-LUTs and 5-LUTs was considered but disregarded because it is extremely inefficient. Adaptive 6-LUTs with varying numbers of extra inputs and degrees of flexibility were investigated, a solution that showed promise. Finally, the adaptive LUT was enhanced to have the ability to share LUT-masks between functions, resulting in the final design of the 8-input fracturable LUT in the ALM shown in [Figure 1](#).

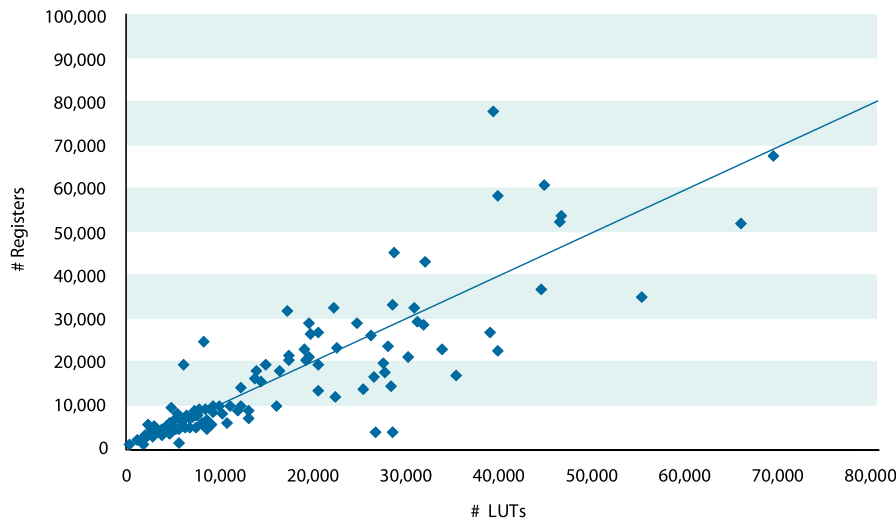
[Figure 6](#) shows a different representation of the ALM in terms of 4-input and 3-input LUTs and multiplexers, illustrating how the LUT-mask can be fractured and shared between two different logic functions. Approximately 150,000 FPGA synthesis, placement, and routing runs were performed to determine the most cost-effective structure that would yield the performance improvement of a 6-LUT.

Figure 6. Adaptive Logic Module (ALM) Block Diagram



In addition to the adaptive LUT, the ALM also contains two registers and two adders as shown in [Figures 1 and 6](#). The extra register was added because experiments indicated that many customer applications required a higher than 1:1 ratio of registers and LUTs. [Figure 7](#) compares the numbers of registers and LUTs in a set of Stratix II customer designs. Almost half of the designs require more registers than LUTs, and will therefore be register-limited in architectures with only one register per combinational block. The ALM has two registers for increased density, creating a superior building block. An extra adder was included to enhance the arithmetic capability of the ALM, allowing for two bits of addition per ALM or a single ternary adder. Thus, the ALM provides twice as much register and arithmetic capability as a basic 6-LUT.

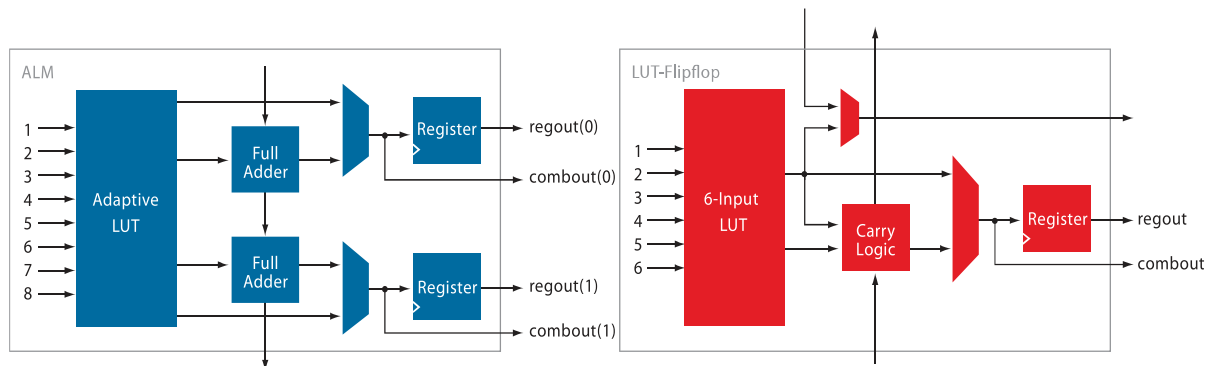
Figure 7. Registers vs. LUTs



The ALM Advantage

The Stratix II ALM has put Altera at least one generation ahead of the competition in FPGA architecture. Introduced more than two years ago, it is significantly more flexible and, as a result, more area efficient than the recently introduced Xilinx Virtex-5 logic element (also called a LUT flipflop pair), which consists of a basic 6-LUT, carry logic, and a single register as shown in Figure 8. In comparison, the combinational logic portion of the ALM has 8 inputs and supports all 6-input functions plus many other combinations of smaller functions using its 2 outputs. The combinational logic portion of the Virtex-5 logic element, a basic 6-LUT, also has 64 bits of CRAM and two outputs like the ALM, but only contains 6 inputs and has a limited ability to implement more than one logic function. One of its outputs is the output of the 6-LUT and the other is the 5-LUT corresponding to the lower half of the configuration RAM.

Figure 8. Comparing the Stratix II ALM and the Virtex-5 LUT-Flipflop Pair



Although the basic 6-LUT has the ability to implement 2 smaller functions, it will usually be used only as a 6-LUT. Because the LUT only has 6 inputs, the required number of shared inputs places severe restrictions on the types of functions that can be combined. These restrictions make using the basic 6-LUT as two 5-LUTs a rare occurrence. In contrast, the 2 additional inputs in the Stratix II ALM allow it to be used as 2 fully functional 5-LUTs, providing a significant area advantage.

Table 2 gives the number of shared inputs required for a few combinations of functions. For example, the ALM can implement 2 independent 4-input functions (no inputs shared), while the Virtex-5 LUT requires 3 shared inputs.

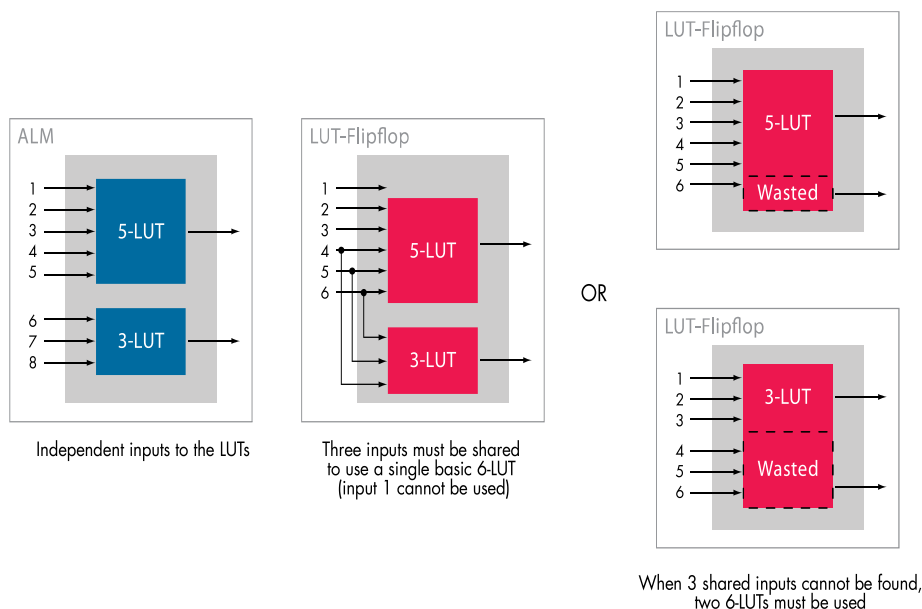
Figure 9 shows another example: the ALM can implement a 5-input and a 3-input function without any shared inputs, while the Virtex-5 LUT requires 3 shared inputs. It is difficult to find functions that can be packed into a Virtex-5 LUT, resulting in functions with less than 6 inputs being implemented in 6-LUT resources.

Table 2. ALM vs. Virtex-5 LUT Flexibility

Output 1	Output 2	Virtex-5	ALM Shared Inputs (Minimum)
5-LUT	5-LUT	5	2
5-LUT	4-LUT	4	1
5-LUT	3-LUT	3	0
4-LUT	4-LUT	3	0
4-LUT	3-LUT	2	0
3-LUT	3-LUT	1	0

Another key advantage of the Stratix II ALM is that each ALM has two registers and two adders as opposed to one register and one adder per Virtex-5 logic element. The result is that, in virtually any design, implementation requires fewer ALMs than Virtex-5 logic elements. See Figure 9.

Figure 9. Implementing 5- and 3-Input Functions in Stratix II ALM and Virtex-5 LUT-Flipflop Pair



Customer Design Benchmarking

To determine how the ALM compares to the Virtex-5 logic element, a benchmarking experiment was run on a set of 65 customer designs. Each design was synthesized using Synplify Pro, once targeting Stratix II devices and once targeting Virtex-5 devices. The synthesized netlists were then run through Quartus II and ISE software with area

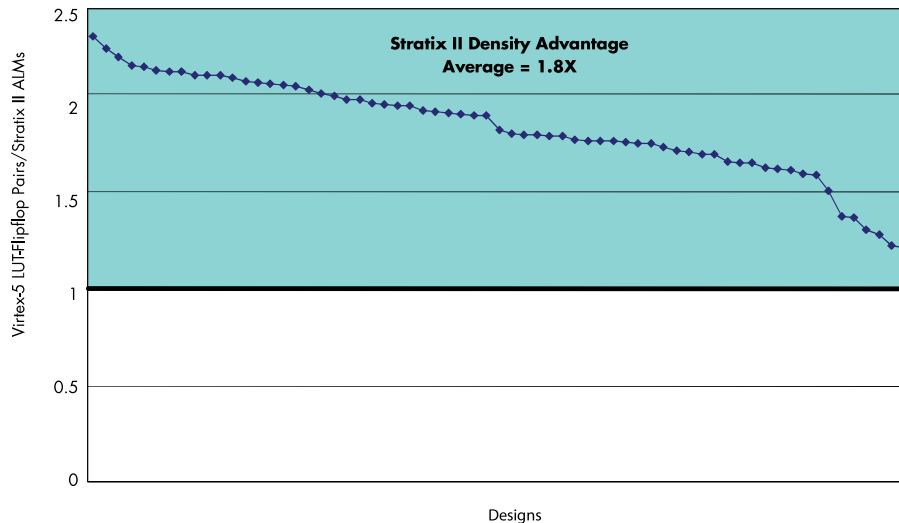
optimization turned on to implement the designs in the smallest number of resources. The versions of the tools are given in [Table 3](#).

Table 3. CAD Tools Used

CAD Tool	Stratix II	Virtex-5
Synthesis	Synplify Pro 8.6.1	Synplify Pro 8.6.1
Placement and Routing	Quartus II 6.0	ISE 8.2i SP1

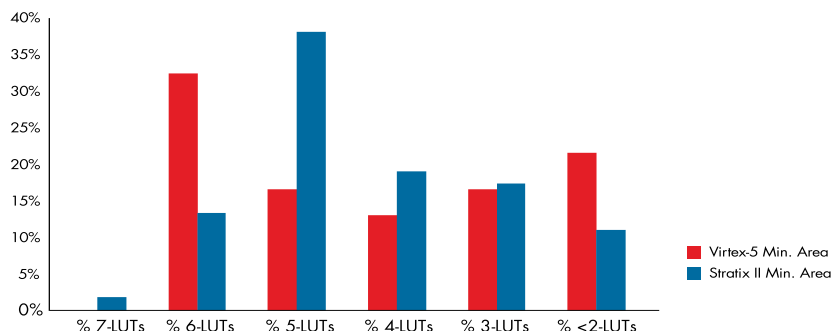
For each implementation of the designs, the resource usage was compared and a ratio of Virtex-5 logic elements (LUT-flipflop pairs) to Stratix II ALMs was computed (see [Figure 10](#)). The black horizontal line labeled "1" indicates the point at which the number of Virtex-5 logic elements and ALMs is the same. Any design above the line shows that more Virtex-5 logic elements than ALMs are required and indicates a Stratix II density advantage. While the ratio of Virtex-5 logic elements to Stratix II ALMs is as high as 2.3, the average across all designs is 1.8X, meaning that one Stratix II ALM is equivalent to (i.e., can hold as much logic as) 1.8 Virtex-5 logic elements.

Figure 10. Density Results



To better understand these results, a breakdown of LUT sizes generated by synthesis is shown in [Figure 11](#). The graph clearly illustrates that synthesis generates a much larger percentage of 6-LUTs for Virtex-5 devices than for Stratix II devices, 32% versus 13%. The reason for this difference is that, when using a basic 6-LUT such as Virtex-5, it is desirable to use as many inputs as possible because the entire LUT is used in most cases regardless of whether the function requires 6 inputs or fewer. Since Virtex-5 can only efficiently implement 6-LUTs, synthesis attempts to generate as many 6-LUTs as possible. Attempting to create smaller functions does not make sense because it is unlikely that two can be packed given the number of inputs that need to be shared.

Figure 11. LUT Sizes Generated During Synthesis



Because the ALM is much more flexible, the synthesis tool can alter the distribution of LUT sizes to produce the right mix of large and small functions, and will result in fewer ALMs being used. Specifically, any function of 5 or fewer inputs uses only half the ALM, making it more important to use the 6-input functions only for speed-critical logic.

Considering only combinational logic, Figure 11 shows that:

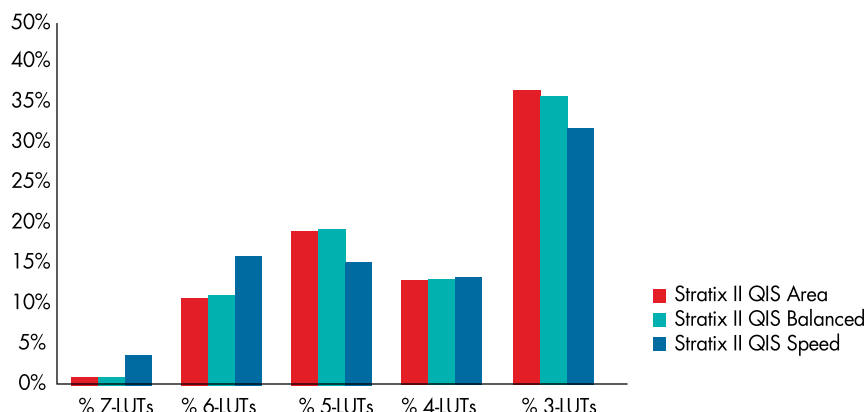
- Stratix II implementations of the designs use only 15% 6- and 7-LUTs combined, each of which will require an ALM.
- The remaining 85% of LUTs (5-LUTs and smaller) each require only a half ALM and effectively fit into 42.5% of the total LUT count, giving the ALM a density advantage over the basic 6-LUT.

Thus, only $15\% + 42.5\% = 57.5\%$ ALMs should be required compared to Virtex-5, or a 1.74X density advantage, while still implementing the speed-critical logic in the 6-LUTs. However, the basic 6-LUT can occasionally pack 2 functions (less than 10% of the time), so the ALM combinational logic density advantage is 1.6X.

For designs with many flipflops, the ALM has a 2:1 density advantage compared to the basic 6-LUT. Therefore, we expect the ALM to be between 1.6X and 2X as dense as the basic 6-LUT. The benchmarking experiment has shown an ALM density advantage of 1.8X, well within this range.

The ALM provides flexibility in software optimization. Figure 12 shows three distributions of LUT sizes produced by Quartus II integrated synthesis (QIS) when optimizing for three different goals: speed, area, or a balanced approach. The mixture of LUT sizes varies depending on the goal. When optimizing for speed, the largest number of 6-LUTs is generated; when optimizing for area, a different distribution that packs in the smallest number of ALMs is generated. This flexibility is unique to Altera and is the result of an intensive research effort on the interaction between software and hardware during architecture development to achieve optimal results.

Figure 12. Distributions of Functions Generated by Quartus II Integrated Synthesis (QIS)

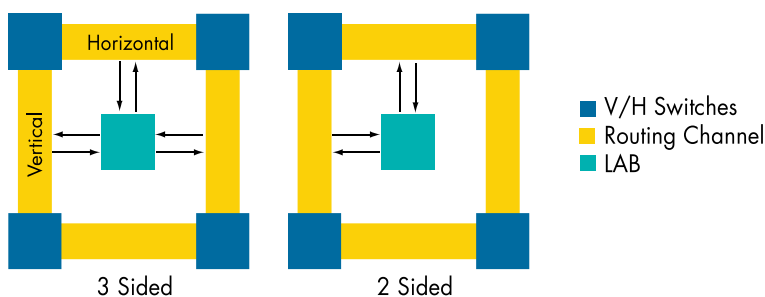


Routing Architecture

In addition to logic block architecture, another key FPGA feature is its routing architecture. The Stratix series of devices introduced the MultiTrack interconnect to maximize connectivity and performance. The routing architecture provides the connectivity between different clusters of logic blocks, called logic array blocks (LABs), and can be measured by the number of "hops" required to get from one LAB to another. The fewer the number of hops and more predictable the pattern, the better the performance and the easier it is for CAD tool optimization.

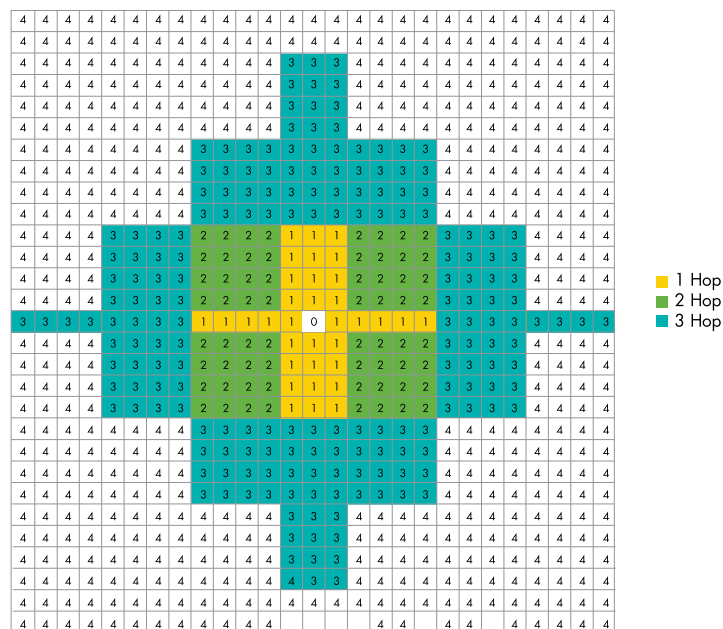
Routing is organized as wires in a number of rows and columns. The Stratix and Stratix II families use a three-sided routing architecture as shown in Figure 13. This means that a LAB can drive or listen to all of the wires on one horizontal (H) channel above it and two vertical (V) channels to the left and right side of it. The channels contain wires of length 4, 8, 16, and 24, and signals can get off at any LAB along the length of the wire.

Figure 13. Number of Routing Architecture Sides



Considering only wires of length four for simplicity, Figure 14 shows the number of hops required to connect to LABs from a given LAB located at the location denoted by '0'.

Figure 14. Stratix and Stratix II Connectivity



The Virtex architectures use a 2-sided routing architecture because a configurable logic block (CLB) can connect to all of the wires in a single vertical channel and a single horizontal channel (with connectivity to half of the wires above and to half of the wires below the CLB). In addition, it uses wires that can only connect to CLBs at select points along the length of the wire. Both of these factors place restrictions on connectivity and placement. With Virtex-5 devices, a CLB can still talk to two channels, but has also included L-shaped (referred to in Xilinx material as diagonal) wires to improve connectivity.

Table 4 compares the connectivity of the Stratix II family with Virtex-5 in terms of the number of LABs/CLB reachable in a given number of hops. In Stratix II devices, many more LABs (34) can be reached in one hop than CLBs in Virtex-5 devices. If the numbers are scaled by the greater efficiency of the ALM, the results are even more favorable to Stratix II devices. Because a LAB contains the equivalent of 20 4-LUT-based LEs versus the ~11 of Virtex-5 (using the 1.8X factor), if we scale the amount of logic that can be reached within a given number of hops by these factors, the improved routing connectivity in terms of logic capacity is even greater.

Table 4. Stratix vs. Virtex Series Connectivity

Hops	Number of LABs/CLBs Reachable		Number of LEs Reachable		Ratio of Stratix II LEs to Virtex-5 LEs
	Stratix II	Virtex-5	Stratix II	Virtex-5	
1	34	12	680	132	5.2
2	96	96	1,920	1,056	1.8
3	160	180	3,200	1,980	1.6
Total	290	288	5,800	3,168	1.8

Conclusion

Altera FPGA architecture is unmatched in the industry and is at least one generation ahead of the competition in terms of logic architecture and two generations ahead in terms of routing architecture. The ALM's ability to divide the combinational logic portion and the availability of eight inputs allow it to implement, in addition to a full 6-input LUT, a variety of smaller functions. The Stratix series of families with a 3-sided routing architecture and wires that can connect to any LAB along their length provides the most connectivity in terms of the amount of logic that can be reached in least number of hops. Both of these accomplishments are the result of numerous innovations made possible through Altera's unique approach and experimental setup to develop superior architectures.

References

Ahmed, Elias and Jonathan Rose. "The Effect of LUT and Cluster Size on Deep-Submicron FPGA Performance and Density." FPGA 2000: ACM Symposium on FPGAs, 3-12. February 2000.

Lewis, D., et al. "The Stratix II Logic and Routing Architecture." FPGA 2005: ACM Symposium on FPGAs, 14-20. February 2005.



101 Innovation Drive
San Jose, CA 95134
(408) 544-7000
<http://www.altera.com>

Copyright © 2006 Altera Corporation. All rights reserved. Altera, The Programmable Solutions Company, the stylized Altera logo, specific device designations, and all other words and logos that are identified as trademarks and/or service marks are, unless noted otherwise, the trademarks and service marks of Altera Corporation in the U.S. and other countries. All other product or service names are the property of their respective holders. Altera products are protected under numerous U.S. and foreign patents and pending applications, maskwork rights, and copyrights. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera Corporation. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.